In the Claims

The status of claims in the case is as follows:

1. [Currently amended] A scalable system for providing a web processing tool, comprising:

a browser;

a plurality of first clustered servers running first mirror image system and application code;

a plurality of second clustered servers running second mirror image system and application code, servers within said second clustered servers periodically replicating with each other so as to maintain data consistency between them;

a database server;

a first network dispatcher for dynamically balancing client workload by redirecting clients to one of said first clustered servers based on current workload of servers within said plurality of first clustered servers;

a second network dispatcher responsive said first clustered servers for dynamically balancing client workload by redirecting clients to one of said second clustered servers based on current workload of servers within said plurality of second clustered servers;

an application server asynchronously responsive to said second clustered servers for running agents to process application data requests and bridge said data with respect to said database server and other back end servers.

2. [Original] The scalable system of claim 1, said first clustered servers being operable for presenting a graphical user interface to the said browser and for caching data on behalf of an end user.

3. [Original] The scalable system of claim 1, said first clustered servers being domino.go servers operable for presenting a graphical user interface to said browser and redirecting said client via said second network dispatcher

to a second cluster server.

4. [Original] The scalable system of claim 1, said web processing tool being a web requisition catalog application.

5. [Original] The scalable system of claim 1, said second clustered servers being operable for performing workflow, providing security, and serving as a document repository.

6. [Original] The scalable system of claim 5, said second clustered servers being domino network servers.

7. [Original] The scalable system of claim 6, said document repository being requisitions stored in domino .nsf files.

8. [Original] The scalable system of claim 2, further comprising an external objects dynamic file for storing external objects in one place for dynamic access by said first clustered servers, and for generating said gui.

9. [Original] The scalable system of claim 1, said database server being a relational database server.

10. [Original] The scalable system of claim 1, said other back end server comprising an enterprise resource planning system, including an accounting application having an accounts payable function.

11. [Previously amended] The scalable system of claim 3, further comprising a configuration file of proxy statements for mapping user requests to said second cluster.

12. [Currently amended] A method for generating on-line procurement requisitions, comprising the steps of:

receiving a client request;

dynamically balancing client workload among servers by directing said request to a first server within a first cluster of virtual servers based on current server workload , each server in said first cluster running first same application and system code;

operating said first server to determine the mapping of said client request and the function required;

responsive to a database access function, dynamically balancing client workload among servers directing said client request to a second server within a second cluster of virtual servers based on current server workload, each server in said second cluster running second same application and system code, servers within said second clustered servers periodically replicating with each other so as to maintain data consistency between them; and

operating said second server to direct said client request to an application server where all data is replicated and where bridges and agents execute with respect to data in said database.

13. [Original] The method of claim 12, further comprising the steps of:

synchronizing all virtual servers within said first cluster; and

synchronizing all virtual servers within said second cluster.

14. [Original] The method of claim 13, further comprising the steps of:

replicating application data to a back-end relational database server; and

replicating application data to a back-end enterprise resource planning system including an accounting application having an accounts payable function.

15. [Previously amended] A program storage device readable by a machine, tangibly embodying a program of instructions executable by a machine to perform method steps for processing a client request with respect to a database, said method steps comprising:

receiving a client request;

dynamically balancing server workload by directing said request to a first server within a first cluster of virtual servers based on current server usage, each server within said first cluster executing first same code;

operating said first server to determine the mapping of said client request and the function required;

responsive to a database access function, dynamically balancing server workload by directing said client request to a second server within a second cluster of virtual servers based on current server usage, each server within said second cluster executing second same code, servers within said second cluster periodically replicating with each other so as to maintain data consistency between them; and

operating said second server to direct said client request to an application server where all data is replicated and where bridges and agents execute with respect to data in said database.

16. [Previously amended] A computer program product or computer program element for:

receiving a client request;

dynamically balancing server workload by directing said request to a first server within a first cluster of virtual servers based on current server usage, each server within said first cluster executing first same code;

operating said first server to determine the mapping of said client request and the function required;

responsive to a database access function, dynamically balancing server workload by directing said client request to a second server within a second cluster of virtual servers based on current server usage, each server within said second cluster executing second same code, servers within said second cluster periodically replicating with each other so as to maintain data consistency between them; and

operating said second server to direct said client request to an application server where all data is replicated and where bridges and agents execute with respect to data in said database.

17. [Previously submitted] The program storage device of claim 15, said method further comprising the step of operating said first clustered servers for presenting a

graphical user interface to the said browser and for caching data on behalf of an end user.

18. [Previously submitted] The program storage device of claim 15, said method further comprising:

synchronizing all virtual servers within said first cluster; and

synchronizing all virtual servers within said second cluster.

19. [Previously submitted] The program storage device of claim 18, said method further comprising:

replicating application data to a back-end relational database server; and

replicating application data to a back-end enterprise resource planning system including an accounting application having an accounts payable function.

20. [Previously submitted] The program storage device of claim 15, said method further comprising storing external

objects in an external objects dynamic file for dynamic access by said first cluster of servers, and for generating said gui.